

ANNEX B (normative) C++ application programmer's interface

```
//File RTI.hh

#ifndef RTI_hh
#define RTI_hh

#include <iostream.h>
#include <math.h>
#include <rw/rwfile.h>

struct RTIambPrivateRefs;
struct RTIambPrivateData;

class RTI {
public:

#include "baseTypes.hh"
#include "RTItypes.hh"

    class RTIambassador {
    public:
#include "RTIambServices.hh"
        RTIambPrivateData* privateData;
    private:
        RTIambPrivateRefs* privateRefs;
    };

    class FederateAmbassador {
    public:
#include "federateAmbServices.hh"
    };
};

#endif
```

```

//File baseTypes.hh
//Included in RTI.hh

#ifndef NULL
#define NULL (0)
#endif

typedef unsigned short UShort;
typedef short Short;
typedef unsigned long ULONG;
typedef long Long;
typedef double Double;
typedef float Float;

enum Boolean {
    RTI_FALSE = 0,
    RTI_TRUE};

class Exception {
public:
    RTI::ULONG _serial;
    char *_reason;
    const char *_name;
    Exception (const char *reason);
    Exception (RTI::ULONG serial, const char *reason=NULL);
    Exception (const Exception &toCopy);
    virtual ~Exception ();
    Exception & operator = (const Exception &);
    friend ostream& operator<< (ostream &, Exception *);
};

#define RTI_EXCEPT(A) \
class A : public RTI::Exception {      \
public: \
    static const char *_ex; \
    A (const char *reason) : Exception (reason) { _name = _ex; } \
    A (RTI::ULONG serial, const char *reason=NULL) \
        : Exception (serial, reason) { _name = _ex; } \
    A (const RTI::Exception &toCopy) : Exception(toCopy) { _name = _ex; } \
};

}

```

```

//File RTItypes.hh
//Included in RTI.h

#define MAX_FEDERATION          32
#define MAX_FEDERATE             32
#define MAX_NAME_LENGTH          64
#define MAX_SPACES                10
#define MAX_OBJECT_CLASSES        200
#define MAX_INTERACTION_CLASSES   200
#define MAX_ATTRIBUTES_PER_CLASS  200
#define MAX_PARAMETERS_PER_CLASS  200
#define MAX_DIMENSIONS_PER_SPACE 10
#define DEFAULT_SPACE_NAME        "defaultSpace"
#define MAX_USER_TAG_LENGTH       30
#define RTI_VERSION                ((RTI::ULong) 0xc0000000)
#define MAX_EXTENT                 ((RTI::ULong) 0x40000000)
#define MIN_EXTENT                 ((RTI::ULong) 0x10000000)

RTI_EXCEPT(ArrayIndexOutOfBoundsException)
RTI_EXCEPT(AsynchronousDeliveryAlreadyDisabled)
RTI_EXCEPT(AsynchronousDeliveryAlreadyEnabled)
RTI_EXCEPT(AttributeAcquisitionWasNotRequested)
RTI_EXCEPT(AttributeAcquisitionWasNotCanceled)
RTI_EXCEPT(AttributeAlreadyBeingAcquired)
RTI_EXCEPT(AttributeAlreadyBeingDivested)
RTI_EXCEPT(AttributeAlreadyOwned)
RTI_EXCEPT(AttributeDivestitureWasNotRequested)
RTI_EXCEPT(AttributeNotDefined)
RTI_EXCEPT(AttributeNotKnown)
RTI_EXCEPT(AttributeNotOwned)
RTI_EXCEPT(AttributeNotPublished)
RTI_EXCEPT(ConcurrentAccessAttempted)
RTI_EXCEPT(CouldNotDiscover)
RTI_EXCEPT(CouldNotOpenFED)
RTI_EXCEPT(CouldNotRestore)
RTI_EXCEPT(DeletePrivilegeNotHeld)
RTI_EXCEPT(DimensionNotDefined)
RTI_EXCEPT(EnableTimeConstrainedPending)
RTI_EXCEPT(EnableTimeConstrainedWasNotPending)
RTI_EXCEPT(EnableTimeRegulationPending)
RTI_EXCEPT(EnableTimeRegulationWasNotPending)
RTI_EXCEPT(ErrorReadingFED)
RTI_EXCEPT(EventNotKnown)
RTI_EXCEPT(FederateAlreadyExecutionMember)
RTI_EXCEPT(FederateInternalError)
RTI_EXCEPT(FederateLoggingServiceCalls)
RTI_EXCEPT(FederateNotExecutionMember)
RTI_EXCEPT(FederateOwnsAttributes)
RTI_EXCEPT(FederateWasNotAskedToReleaseAttribute)
RTI_EXCEPT(FederatesCurrentlyJoined)
RTI_EXCEPT(FederationExecutionAlreadyExists)
RTI_EXCEPT(FederationExecutionDoesNotExist)
RTI_EXCEPT(FederationTimeAlreadyPassed)
RTI_EXCEPT(HandleValuePairMaximumExceeded)
RTI_EXCEPT(InteractionClassNotDefined)
RTI_EXCEPT(InteractionClassNotKnown)
RTI_EXCEPT(InteractionClassNotPublished)
RTI_EXCEPT(InteractionClassNotSubscribed)
RTI_EXCEPT(InteractionParameterNotDefined)
RTI_EXCEPT(InteractionParameterNotKnown)
RTI_EXCEPT(InvalidExtents)
RTI_EXCEPT(InvalidFederationTime)
RTI_EXCEPT(InvalidHandleValuePairSetContext)
RTI_EXCEPT(InvalidLookahead)
RTI_EXCEPT(InvalidOrderingHandle)
RTI_EXCEPT(InvalidRegionContext)
RTI_EXCEPT(InvalidResignAction)
RTI_EXCEPT(InvalidRetractionHandle)
RTI_EXCEPT(InvalidTransportationHandle)
RTI_EXCEPT(MemoryExhausted)
RTI_EXCEPT(NameNotFound)
RTI_EXCEPT(ObjectClassNotDefined)
RTI_EXCEPT(ObjectClassNotKnown)
RTI_EXCEPT(ObjectClassNotPublished)

```

```

RTI_EXCEPT(ObjectClassNotSubscribed)
RTI_EXCEPT(ObjectNotKnown)
RTI_EXCEPT(ObjectAlreadyRegistered)
RTI_EXCEPT(RegionNotKnown)
RTI_EXCEPT(RestoreInProgress)
RTI_EXCEPT(RestoreNotRequested)
RTI_EXCEPT(RTIinternalError)
RTI_EXCEPT(SpaceNotDefined)
RTI_EXCEPT(SaveInProgress)
RTI_EXCEPT(SaveNotInitiated)
RTI_EXCEPT(SpecifiedSaveLabelDoesNotExist)
RTI_EXCEPT(SynchronizationPointLabelWasNotAnnounced)
RTI_EXCEPT(TimeAdvanceAlreadyInProgress)
RTI_EXCEPT(TimeAdvanceWasNotInProgress)
RTI_EXCEPT(TimeConstrainedAlreadyEnabled)
RTI_EXCEPT(TimeConstrainedWasNotEnabled)
RTI_EXCEPT(TimeRegulationAlreadyEnabled)
RTI_EXCEPT(TimeRegulationWasNotEnabled)
RTI_EXCEPT(UnableToPerformSave)
RTI_EXCEPT(ValueCountExceeded)
RTI_EXCEPT(ValueLengthExceeded)

enum ObjectState {
    KNOWN_TO_FEDERATE = 1,
    HOLDING_TOKENS,
    DELETED
};

enum ResignAction {
    RELEASE_ATTRIBUTES = 1,
    DELETE_OBJECTS,
    DELETE_OBJECTS_AND_RELEASE_ATTRIBUTES,
    NO_ACTION
};

enum FederateStateType {
    RUNNING = 1,
    SAVING,
    RESTORING
};

enum TimeManagerStateType {
    IDLE = 1,
    TIME_ADVANCING
};

class Region;

class FederateAmbassador;

typedef FederateAmbassador *FederateAmbassadorPtr;

typedef RTI::Long SpaceHandle;

typedef RTI::ULong ObjectClassHandle;

typedef RTI::ULong InteractionClassHandle;

typedef RTI::ULong ExtentIndex;

typedef RTI::ULong Handle;

typedef Handle AttributeHandle;

typedef Handle ParameterHandle;

typedef Handle ObjectHandle;

typedef Handle DimensionHandle;

typedef RTI::ULong FederateHandle;

typedef Handle TransportationHandle;

```

```

typedef TransportationHandle TransportType;
typedef Handle OrderingHandle;
typedef OrderingHandle OrderType;
typedef RTI::ULong FederateID;
typedef RTI::ULong UniqueID;
typedef RTI::Double TickTime;
typedef RTI::ULong RegionToken;

class AttributeHandleValuePairSet {
// Instances of class HandleValuePairSet are the containers used to pass
// object attribute values and interaction parameter values between the
// Federate and the RTI. These containers hold sets of attribute/parameter
// values indexed by their attribute/parameter handle. Instances of this
// class are provided to the RTI in the Update Attribute Values and Send
// Interaction service invocations. Instances of this class are provided
// to the Federate in the Reflect Attribute Values and Receive Interaction
// service invocations. When instances of HandleValuePairSet are provided
// to the Federate by the RTI, the memory used to store attribute/parameter
// values is valid for use by the federate only within the scope of the
// Reflect Attribute Values or Receive Interaction service invocation.
// Symmetrically, for instances of HandleValuePairSet provided by the Federate
// to the RTI, the memory used to store attribute/parameter values is valid for
// use by the RTI only within the scope of the Update Attribute Values or Send
// Interaction service invocation.
public:
    virtual ~AttributeHandleValuePairSet() { ; }

    virtual ULong size() const = 0;

    virtual Handle getHandle(
        RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual ULong getValueLength(
        RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void getValue(
        RTI::ULong i,
        char*       buff,
        ULong&      valueLength) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual char *getValuePointer(
        RTI::ULong i,
        ULong&      valueLength) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual RTI::TransportType getTransportType( RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException,
        InvalidHandleValuePairSetContext) = 0;

    virtual RTI::OrderType getOrderType( RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException,
        InvalidHandleValuePairSetContext) = 0;

    virtual RTI::Region *getRegion(
        RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException,
        InvalidHandleValuePairSetContext) = 0;

```

```

virtual void add(
    Handle      h,
    const char* buff,
    ULong       valueLength)
throw (
    ValueLengthExceeded,
    ValueCountExceeded) = 0;

virtual void remove(// not guaranteed safe while iterating
    Handle      h)
throw (
    ArrayIndexOutOfBoundsException) = 0;

virtual void moveFrom(
    const AttributeHandleValuePairSet& ahvps,
    RTI::ULong&                      i)
throw (
    ValueCountExceeded,
    ArrayIndexOutOfBoundsException) = 0;

virtual void empty() = 0; // Empty the Set without deallocating space.

virtual inline RTI::ULong start() const = 0;
virtual inline RTI::ULong valid(RTI::ULong i) const = 0;
virtual inline RTI::ULong next(RTI::ULong i) const = 0;
};

class AttributeSetFactory {
public:
    static AttributeHandleValuePairSet* create(
        ULong count)
    throw (
        MemoryExhausted,
        ValueCountExceeded,
        HandleValuePairMaximumExceeded);
};

class AttributeHandleSet {
public:
    virtual ~AttributeHandleSet() { ; }

    virtual ULong size() const = 0;

    virtual AttributeHandle getHandle(ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void add(AttributeHandle h)
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void remove(AttributeHandle h)
    throw (// not guaranteed safe while iterating
        AttributeNotDefined) = 0;

    virtual void empty() = 0; // Empty the Set

    virtual Boolean isEmpty() const = 0; //is set empty?
    virtual Boolean isMember(AttributeHandle h) const = 0;
};

class AttributeHandleSetFactory {
public:
    static AttributeHandleSet* create(
        ULong count)
    throw(
        MemoryExhausted,
        ValueCountExceeded);
};

class FederateHandleSet {

```

```

public:
    virtual ULong size() const = 0;

    virtual FederateHandle getHandle(ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void add(FederateHandle h)
    throw (
        ValueCountExceeded) = 0;

    virtual void remove(FederateHandle h)
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void empty() = 0; // Empty the set without deallocating space.

    virtual Boolean isMember(FederateHandle h) const = 0;
};

class FederateHandleSetFactory {
public:
    static FederateHandleSet* create(ULong count)
    throw (
        MemoryExhausted,
        ValueCountExceeded);
};

class ParameterHandleValuePairSet {
// Instances of class HandleValuePairSet are the containers used to pass
// object attribute values and interaction parameter values between the
// Federate and the RTI. These containers hold sets of attribute/parameter
// values indexed by their attribute/parameter handle. Instances of this
// class are provided to the RTI in the Update Attribute Values and Send
// Interaction service invocations. Instances of this class are provided
// to the Federate in the Reflect Attribute Values and Receive Interaction
// service invocations. When instances of HandleValuePairSet are provided
// to the Federate by the RTI, the memory used to store attribute/parameter
// values is valid for use by the federate only within the scope of the
// Reflect Attribute Values or Receive Interaction service invocation.
// Symmetrically, for instances of HandleValuePairSet provided by the Federate
// to the RTI, the memory used to store attribute/parameter values is valid for
// use by the RTI only within the scope of the Update Attribute Values or Send
// Interaction service invocation.
public:
    virtual ~ParameterHandleValuePairSet() { ; }

    virtual ULong size() const = 0;

    virtual Handle getHandle(
        RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual ULong getValueLength(
        RTI::ULong i) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual void getValue(
        RTI::ULong i,
        char*      buff,
        ULong&     valueLength) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual char *getValuePointer(
        RTI::ULong i,
        ULong&     valueLength) const
    throw (
        ArrayIndexOutOfBoundsException) = 0;

    virtual RTI::TransportType getTransportType(void) const

```

```

        throw ( InvalidHandleValuePairSetContext) = 0;

    virtual RTI::OrderType getOrderType(void) const
        throw ( InvalidHandleValuePairSetContext) = 0;

    virtual RTI::Region *getRegion(void) const
        throw ( InvalidHandleValuePairSetContext) = 0;

    virtual void add(
        Handle      h,
        const char* buff,
        ULong       valueLength)
        throw (
            ValueLengthExceeded,
            ValueCountExceeded) = 0;

    virtual void remove(// not guaranteed safe while iterating
        Handle      h)
        throw (
            ArrayIndexOutOfBoundsException) = 0;

    virtual void moveFrom(
        const ParameterHandleValuePairSet& phvps,
        RTI::ULong&                  i)
        throw (
            ValueCountExceeded,
            ArrayIndexOutOfBoundsException) = 0;

    virtual void empty() = 0; // Empty the Set without deallocating space.

    virtual inline RTI::ULong start() const = 0;
    virtual inline RTI::ULong valid(RTI::ULong i) const = 0;
    virtual inline RTI::ULong next(RTI::ULong i) const = 0;
};

class ParameterSetFactory {
public:
    static ParameterHandleValuePairSet* create(ULong count)
        throw (
            MemoryExhausted,
            ValueCountExceeded,
            HandleValuePairMaximumExceeded);
};

class Region {
public:

    virtual ~Region();

    virtual ULong getRangeLowerBound(
        ExtentIndex   theExtent,
        DimensionHandle theDimension) const
        throw (
            ArrayIndexOutOfBoundsException) = 0;

    virtual ULong getRangeUpperBound(
        ExtentIndex   theExtent,
        DimensionHandle theDimension) const
        throw (
            ArrayIndexOutOfBoundsException) = 0;

    virtual void setRangeLowerBound(
        ExtentIndex   theExtent,
        DimensionHandle theDimension,
        ULong         theLowerBound)
        throw (
            ArrayIndexOutOfBoundsException) = 0;

    virtual void setRangeUpperBound(
        ExtentIndex   theExtent,
        DimensionHandle theDimension,
        ULong         theUpperBound)
        throw (
            ArrayIndexOutOfBoundsException) = 0;
};

```

```

virtual SpaceHandle getSpaceHandle() const
throw (
) = 0;

virtual ULong getNumberOfExtents() const
throw (
) = 0;

virtual ULong getRangeLowerBoundNotificationLimit(
ExtentIndex theExtent,
DimensionHandle theDimension) const
throw (
ArrayIndexOutOfBoundsException) = 0;

virtual ULong getRangeUpperBoundNotificationLimit(
ExtentIndex theExtent,
DimensionHandle theDimension) const
throw (
ArrayIndexOutOfBoundsException) = 0;

};

class FedTime {
public:
    virtual ~FedTime();

    virtual void setZero() = 0;
    virtual Boolean isZero() = 0;
    virtual void setEpsilon() = 0;
    virtual void setPositiveInfinity() = 0;
    virtual Boolean isPositiveInfinity() = 0;
    virtual FedTime& operator+=(const FedTime&)
throw (
InvalidFederationTime) = 0;
    virtual FedTime& operator-=(const FedTime&)
throw (
InvalidFederationTime) = 0;
    virtual Boolean operator<= (const FedTime&) const
throw (
InvalidFederationTime) = 0;
    virtual Boolean operator< (const FedTime&) const
throw (
InvalidFederationTime) = 0;
    virtual Boolean operator>= (const FedTime&) const
throw (
InvalidFederationTime) = 0;
    virtual Boolean operator> (const FedTime&) const
throw (
InvalidFederationTime) = 0;
    virtual Boolean operator== (const FedTime&) const
throw (
InvalidFederationTime) = 0;
    virtual FedTime& operator= (const FedTime&)
throw (
InvalidFederationTime) = 0;
    //return bytes needed to encode
    virtual int encodedLength() const = 0;
    //encode into supplied buffer
    virtual void encode(char *buff) const = 0;
};

```

```
virtual int getPrintableLength() const = 0;  
virtual void getPrintableString(char*) = 0;  
};  
  
class FedTimeFactory {  
public:  
    static FedTime* makeZero()  
        throw (  
            MemoryExhausted);  
  
    static FedTime* decode(const char *buf)  
        throw (  
            MemoryExhausted);  
};  
  
struct EventRetractionHandle_s {  
    UniqueID          theSerialNumber;  
    FederateHandle   sendingFederate;  
};  
typedef struct EventRetractionHandle_s EventRetractionHandle;
```

```

//File RTIambServices.hh
//Included in RTI.hh

// RTI Parameter Passing Memory Conventions
//
// C1 In parameter by value.
// C2 Out parameter by pointer value.
// C3 Function return by value.
// C4 In parameter by const pointer value. Caller provides memory.
// Caller may free memory or overwrite it upon completion of
// the call. Callee must copy during the call anything it
// wishes to save beyond completion of the call. Parameter
// type must define const accessor methods.
// C5 Out parameter by pointer value. Caller provides reference to object.
// Callee constructs an instance on the heap (new) and returns.
// The caller destroys the instance (delete) at its leisure.
// C6 Function return by pointer value. Callee constructs an instance on
// the heap (new) and returns a reference. The caller destroys the
// instance (delete) at its leisure.
//

typedef FederateAmbassador *FederateAmbassadorPtr;

///////////////////////////////
// Federation Management Services //
///////////////////////////////

// 4.2
void createFederationExecution (
    const char *executionName, // supplied C4
    const char *FED)          // supplied C4
throw (
    FederationExecutionAlreadyExists,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 4.3
void destroyFederationExecution (
    const char *executionName) // supplied C4
throw (
    FederatesCurrentlyJoined,
    FederationExecutionDoesNotExist,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 4.4
FederateHandle
joinFederationExecution (                                // returned C3
    const char           *yourName,                      // supplied C4
    const char           *executionName,                  // supplied C4
    FederateAmbassadorPtr federateAmbassadorReference) // supplied C1
throw (
    FederateAlreadyExecutionMember,
    FederationExecutionDoesNotExist,
    CouldNotOpenFED,
    ErrorReadingFED,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 4.5
void resignFederationExecution (
    ResignAction theAction) // supplied C1
throw (
    FederateOwnsAttributes,
    FederateNotExecutionMember,
    InvalidResignAction,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 4.6
void registerFederationSynchronizationPoint (

```

```

const char *label, // supplied C4
const char *theTag) // supplied C4
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void registerFederationSynchronizationPoint (
    const char *label, // supplied C4
    const char *theTag, // supplied C4
    const FederateHandleSet& syncSet) // supplied C4
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 4.9
void synchronizationPointAchieved (
    const char *label) // supplied C4
throw (
    SynchronizationPointLabelWasNotAnnounced,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 4.11
void requestFederationSave (
    const char *label, // supplied C4
    const FedTime& theTime) // supplied C4
throw (
    FederationTimeAlreadyPassed,
    InvalidFederationTime,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void requestFederationSave (
    const char *label) // supplied C4
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 4.13
void federateSaveBegun ()
throw (
    SaveNotInitiated,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RestoreInProgress,
    RTIinternalError);

// 4.14
void federateSaveComplete ()
throw (
    SaveNotInitiated,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RestoreInProgress,
    RTIinternalError);

void federateSaveNotComplete ()
throw (
    SaveNotInitiated,

```

```

FederateNotExecutionMember,
ConcurrentAccessAttempted,
RestoreInProgress,
RTIinternalError);

// 4.16
void requestFederationRestore (
    const char *label) // supplied C4
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 4.20
void federateRestoreComplete ()
throw (
    RestoreNotRequested,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

void federateRestoreNotComplete ()
throw (
    RestoreNotRequested,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

///////////////////////////////
// Declaration Management Services //
///////////////////////////////

// 5.2
void publishObjectClass (
    ObjectClassHandle theClass,           // supplied C1
    const AttributeHandleSet& attributeList) // supplied C4
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.3
void unpublishObjectClass (
    ObjectClassHandle theClass) // supplied C1
throw (
    ObjectClassNotDefined,
    ObjectClassNotPublished,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.4
void publishInteractionClass (
    InteractionClassHandle theInteraction) // supplied C1
throw (
    InteractionClassNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.5
void unpublishInteractionClass (
    InteractionClassHandle theInteraction) // supplied C1
throw (

```

```

InteractionClassNotDefined,
InteractionClassNotPublished,
FederateNotExecutionMember,
ConcurrentAccessAttempted,
SaveInProgress,
RestoreInProgress,
RTIinternalError);

// 5.6
void subscribeObjectClassAttributes (
    ObjectClassHandle theClass,           // supplied C1
    const AttributeHandleSet& attributeList, // supplied C4
    RTI::Boolean active = RTI_TRUE)
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.7
void unsubscribeObjectClass (
    ObjectClassHandle theClass) // supplied C1
throw (
    ObjectClassNotDefined,
    ObjectClassNotSubscribed,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.8
void subscribeInteractionClass (
    InteractionClassHandle theClass, // supplied C1
    RTI::Boolean active = RTI_TRUE)
throw (
    InteractionClassNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    FederateLoggingServiceCalls,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 5.9
void unsubscribeInteractionClass (
    InteractionClassHandle theClass) // supplied C1
throw (
    InteractionClassNotDefined,
    InteractionClassNotSubscribed,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

///////////////////////////////
// Object Management Services //
///////////////////////////////

// 6.2
ObjectHandle                                // returned C3
registerObjectInstance (
    ObjectClassHandle theClass, // supplied C1
    const char *theObject) // supplied C4
throw (
    ObjectClassNotDefined,
    ObjectClassNotPublished,
    ObjectAlreadyRegistered,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    
```

```

SaveInProgress,
RestoreInProgress,
RTIinternalError);

ObjectHandle                               // returned C3
registerObjectInstance (
    ObjectClassHandle theClass) // supplied C1
throw (
    ObjectClassNotDefined,
    ObjectClassNotPublished,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 6.4
EventRetractionHandle                      // returned C3
updateAttributeValue (
    ObjectHandle           theObject,      // supplied C1
    const AttributeHandleValuePairSet& theAttributes, // supplied C4
    const FedTime&          theTime,        // supplied C4
    const char*              *theTag)       // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    InvalidFederationTime,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void updateAttributeValue (
    ObjectHandle           theObject,      // supplied C1
    const AttributeHandleValuePairSet& theAttributes, // supplied C4
    const char*              *theTag)       // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 6.6
EventRetractionHandle                      // returned C3
sendInteraction (
    InteractionClassHandle   theInteraction, // supplied C1
    const ParameterHandleValuePairSet& theParameters, // supplied C4
    const FedTime&            theTime,        // supplied C4
    const char*               *theTag)       // supplied C4
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InteractionParameterNotDefined,
    InvalidFederationTime,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void sendInteraction (
    InteractionClassHandle   theInteraction, // supplied C1
    const ParameterHandleValuePairSet& theParameters, // supplied C4
    const char*               *theTag)       // supplied C4
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InteractionParameterNotDefined,

```

```

FederateNotExecutionMember,
ConcurrentAccessAttempted,
SaveInProgress,
RestoreInProgress,
RTIinternalError);

// 6.8
EventRetractionHandle           // returned C3
deleteObjectInstance (
    ObjectHandle   ObjectHandle, // supplied C1
    const FedTime&   theTime,     // supplied C4
    const char      *theTag)    // supplied C4
throw (
    ObjectNotKnown,
    DeletePrivilegeNotHeld,
    InvalidFederationTime,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void deleteObjectInstance (
    ObjectHandle   ObjectHandle, // supplied C1
    const char      *theTag)    // supplied C4
throw (
    ObjectNotKnown,
    DeletePrivilegeNotHeld,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 6.10
void localDeleteObjectInstance (
    ObjectHandle   ObjectHandle) // supplied C1
throw (
    ObjectNotKnown,
    FederateOwnsAttributes,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 6.11
void changeAttributeTransportType (
    ObjectHandle   theObject,      // supplied C1
    const AttributeHandleSet&   theAttributes, // supplied C4
    TransportationHandle   theType)    // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    InvalidTransportationHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 6.12
void changeInteractionTransportType (
    InteractionClassHandle theClass, // supplied C1
    TransportationHandle   theType) // supplied C1
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InvalidTransportationHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

```

```

RTIinternalError);

// 6.15
void requestObjectAttributeValueUpdate (
    ObjectHandle          theObject,      // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void requestClassAttributeValueUpdate (
    ObjectClassHandle     theClass,        // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

///////////////////////////////
// Ownership Management Services //
///////////////////////////////

// 7.2
void unconditionalAttributeOwnershipDivestiture (
    ObjectHandle          theObject,      // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.3
void negotiatedAttributeOwnershipDivestiture (
    ObjectHandle          theObject,      // supplied C1
    const AttributeHandleSet& theAttributes, // supplied C4
    const char              *theTag)       // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    AttributeAlreadyBeingDivested,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.7
void attributeOwnershipAcquisition (
    ObjectHandle          theObject,      // supplied C1
    const AttributeHandleSet& desiredAttributes, // supplied C4
    const char              *theTag)       // supplied C4
throw (
    ObjectNotKnown,
    ObjectClassNotPublished,
    AttributeNotDefined,
    AttributeNotPublished,
    FederateOwnsAttributes,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,

```

```

SaveInProgress,
RestoreInProgress,
RTIinternalError);

// 7.8
void attributeOwnershipAcquisitionIfAvailable (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& desiredAttributes) // supplied C4
throw (
    ObjectNotKnown,
    ObjectClassNotPublished,
    AttributeNotDefined,
    AttributeNotPublished,
    FederateOwnsAttributes,
    AttributeAlreadyBeingAcquired,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.11
AttributeHandleSet*                      // returned C6
attributeOwnershipReleaseResponse (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    FederateWasNotAskedToReleaseAttribute,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.12
void cancelNegotiatedAttributeOwnershipDivestiture (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    AttributeDivestitureWasNotRequested,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.13
void cancelAttributeOwnershipAcquisition (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeAlreadyOwned,
    AttributeAcquisitionWasNotRequested,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 7.15
void queryAttributeOwnership (
    ObjectHandle      theObject,          // supplied C1
    AttributeHandle   theAttribute) // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotDefined,

```

```

FederateNotExecutionMember,
ConcurrentAccessAttempted,
SaveInProgress,
RestoreInProgress,
RTIinternalError);

// 7.17
Boolean                      // returned C3
isAttributeOwnedByFederate (
    ObjectHandle theObject,   // supplied C1
    AttributeHandle theAttribute) // supplied C1
throw (
    ObjectNotFound,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

///////////////////////////////
// Time Management Services //
///////////////////////////////

// 8.2
void enableTimeRegulation (
    const FedTime& theFederateTime, // supplied C4
    const FedTime& theLookahead) // supplied C4
throw (
    TimeRegulationAlreadyEnabled,
    EnableTimeRegulationPending,
    TimeAdvanceAlreadyInProgress,
    InvalidFederationTime,
    InvalidLookahead,
    ConcurrentAccessAttempted,
    FederateNotExecutionMember,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.4
void disableTimeRegulation ()
throw (
    TimeRegulationWasNotEnabled,
    ConcurrentAccessAttempted,
    FederateNotExecutionMember,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.5
void enableTimeConstrained ()
throw (
    TimeConstrainedAlreadyEnabled,
    EnableTimeConstrainedPending,
    TimeAdvanceAlreadyInProgress,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.7
void disableTimeConstrained ()
throw (
    TimeConstrainedWasNotEnabled,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.8
void timeAdvanceRequest (

```

```

const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    FederationTimeAlreadyPassed,
    TimeAdvanceAlreadyInProgress,
    EnableTimeRegulationPending,
    EnableTimeConstrainedPending,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.9
void timeAdvanceRequestAvailable (
const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    FederationTimeAlreadyPassed,
    TimeAdvanceAlreadyInProgress,
    EnableTimeRegulationPending,
    EnableTimeConstrainedPending,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.10
void nextEventRequest (
    const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    FederationTimeAlreadyPassed,
    TimeAdvanceAlreadyInProgress,
    EnableTimeRegulationPending,
    EnableTimeConstrainedPending,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.11
void nextEventRequestAvailable (
    const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    FederationTimeAlreadyPassed,
    TimeAdvanceAlreadyInProgress,
    EnableTimeRegulationPending,
    EnableTimeConstrainedPending,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.12
void flushQueueRequest (
    const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    FederationTimeAlreadyPassed,
    TimeAdvanceAlreadyInProgress,
    EnableTimeRegulationPending,
    EnableTimeConstrainedPending,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.14

```

```

void enableAsynchronousDelivery()
throw (
    AsynchronousDeliveryAlreadyEnabled,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.15
void disableAsynchronousDelivery()
throw (
    AsynchronousDeliveryAlreadyDisabled,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.16
void queryLBTS (
    FedTime& theTime) // returned C5
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.17
void queryFederateTime (
    FedTime& theTime) // returned C5
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.18
void queryMinNextEventTime (
    FedTime& theTime) // returned C5
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.19
void modifyLookahead (
    const FedTime& theLookahead) // supplied C4
throw (
    InvalidLookahead,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.20
void queryLookahead (
    FedTime& theTime) // returned C5
throw (
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.21
void retract (
    EventRetractionHandle theHandle) // supplied C1
throw (

```

```

    InvalidRetractionHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.23
void changeAttributeOrderType (
    ObjectHandle          theObject,      // supplied C1
    const AttributeHandleSet& theAttributes, // supplied C4
    OrderingHandle         theType)        // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    AttributeNotOwned,
    InvalidOrderingHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 8.24
void changeInteractionOrderType (
    InteractionClassHandle theClass, // supplied C1
    OrderingHandle          theType) // supplied C1
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InvalidOrderingHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

///////////////////////////////
// Data Distribution Management //
///////////////////////////////

// 9.2
Region* createRegion (                                // returned C6
    SpaceHandle theSpace,           // supplied C1
    ULONG       numberofExtents)   // supplied C1
throw (
    SpaceNotDefined,
    InvalidExtents,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.3
void notifyAboutRegionModification (
    Region &theRegion) // supplied C4
throw (
    RegionNotKnown,
    InvalidExtents,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.4
void deleteRegion (
    Region *theRegion) // supplied C1
throw (
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,

```

```

SaveInProgress,
RestoreInProgress,
RTIinternalError);

// 9.5
ObjectHandle registerObjectInstanceWithRegion ( // returned C3
    ObjectClassHandle theClass, // supplied C1
    const char *theObject, // supplied C4
    AttributeHandle theAttributes[], // supplied C4
    Region *theRegions[], // supplied C4
    ULONG theNumberOfHandles) // supplied C1
throw (
    ObjectClassNotDefined,
    ObjectClassNotPublished,
    ObjectAlreadyRegistered,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

ObjectHandle registerObjectInstanceWithRegion ( // returned C3
    ObjectClassHandle theClass, // supplied C1
    AttributeHandle theAttributes[], // supplied C4
    Region *theRegions[], // supplied C4
    ULONG theNumberOfHandles) // supplied C1
throw (
    ObjectClassNotDefined,
    ObjectClassNotPublished,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.6
void associateRegionForUpdates (
    Region &theRegion, // supplied C4
    ObjectHandle theObject, // supplied C1
    const AttributeHandleSet &theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotDefined,
    InvalidRegionContext,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.7
void unassociateRegionForUpdates (
    Region &theRegion, // supplied C4
    ObjectHandle theObject) // supplied C1
throw (
    ObjectNotKnown,
    InvalidRegionContext,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.8
void subscribeObjectClassAttributesWithRegion (
    ObjectClassHandle theClass, // supplied C1
    Region &theRegion, // supplied C4
    const AttributeHandleSet &attributeList, // supplied C4

```

```

        RTI::Boolean      active = RTI_TRUE)
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.9
void unsubscribeObjectClassWithRegion (
    ObjectClassHandle theClass,           // supplied C1
    Region          &theRegion)           // supplied C4
throw (
    ObjectClassNotDefined,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.10
void subscribeInteractionClassWithRegion (
    InteractionClassHandle theClass,       // supplied C1
    Region          &theRegion,           // supplied C4
    RTI::Boolean      active = RTI::RTI_TRUE)
throw (
    InteractionClassNotDefined,
    RegionNotKnown,
    FederateLoggingServiceCalls,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.11
void unsubscribeInteractionClassWithRegion (
    InteractionClassHandle theClass, // supplied C1
    Region          &theRegion) // supplied C4
throw (
    InteractionClassNotDefined,
    InteractionClassNotSubscribed,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.12
EventRetractionHandle                  // returned C3
sendInteractionWithRegion (
    InteractionClassHandle   theInteraction, // supplied C1
    const ParameterHandleValuePairSet &theParameters, // supplied C4
    const FedTime&               theTime,           // supplied C4
    const char                 *theTag,            // supplied C4
    const Region                &theRegion)         // supplied C4
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InteractionParameterNotDefined,
    InvalidFederationTime,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

void sendInteractionWithRegion (

```

```

        InteractionClassHandle      theInteraction, // supplied C1
const ParameterHandleValuePairSet &theParameters, // supplied C4
const char                      *theTag,       // supplied C4
const Region                    &theRegion)    // supplied C4
throw (
    InteractionClassNotDefined,
    InteractionClassNotPublished,
    InteractionParameterNotDefined,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 9.13
void requestClassAttributeValueUpdateWithRegion (
    ObjectClassHandle theClass,           // supplied C1
    const AttributeHandleSet &theAttributes, // supplied C4
    const Region             &theRegion)     // supplied C4
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    RegionNotKnown,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

///////////////////////////////
// RTI Support Services //
////////////////////////////

// 10.2
ObjectClassHandle // returned C3
getObjectClassHandle (
    const char *theName) // supplied C4
throw (
    NameNotFound,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.3
char *                  // returned C6
getObjectName (
    ObjectClassHandle theHandle) // supplied C1
throw (
    ObjectClassNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.4
AttributeHandle          // returned C3
getAttributeHandle (
    const char          *theName,      // supplied C4
    ObjectClassHandle  whichClass)   // supplied C1
throw (
    ObjectClassNotDefined,
    NameNotFound,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.5
char *                  // returned C6
getAttributeName (
    AttributeHandle  theHandle,    // supplied C1
    ObjectClassHandle whichClass) // supplied C1
throw (
    ObjectClassNotDefined,
    AttributeNotDefined,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

```

```

FederateNotExecutionMember,
ConcurrentAccessAttempted,
RTIinternalError);

// 10.6
InteractionClassHandle      // returned C3
getInteractionClassHandle (
  const char *theName)      // supplied C4
throw (
  NameNotFound,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.7
char *                      // returned C6
getInteractionClassName (
  InteractionClassHandle theHandle) // supplied C1
throw (
  InteractionClassNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.8
ParameterHandle             // returned C3
getParameterHandle (
  const char *theName,        // supplied C4
  InteractionClassHandle whichClass) // supplied C1
throw (
  InteractionClassNotDefined,
  NameNotFound,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.9
char *                      // returned C6
getParameterName (
  ParameterHandle theHandle, // supplied C1
  InteractionClassHandle whichClass) // supplied C1
throw (
  InteractionClassNotDefined,
  InteractionParameterNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.10
ObjectHandle                // returned C3
getObjectInstanceHandle (
  const char *theName)      // supplied C4
throw (
  ObjectNotKnown,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.11
char *                      // returned C6
getObjectInstanceName (
  ObjectHandle theHandle) // supplied C1
throw (
  ObjectNotKnown,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.12
SpaceHandle                 // returned C3
getRoutingSpaceHandle (
  const char *theName)      // supplied C4
throw (
  NameNotFound,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

```

```

FederateNotExecutionMember,
ConcurrentAccessAttempted,
RTIinternalError);

// 10.13
char *                      // returned C6
getRoutingSpaceName (
  const SpaceHandle theHandle) // supplied C4
throw (
  SpaceNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.14
DimensionHandle              // returned C3
getDimensionHandle (
  const char      *theName,    // supplied C4
  SpaceHandle     whichSpace) // supplied C1
throw (
  SpaceNotDefined,
  NameNotFound,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.15
char *                      // returned C6
getDimensionName (
  DimensionHandle theHandle, // supplied C1
  SpaceHandle      whichSpace) // supplied C1
throw (
  SpaceNotDefined,
  DimensionNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.16
SpaceHandle                  // returned C3
getAttributeRoutingSpaceHandle (
  AttributeHandle   theHandle, // supplied C1
  ObjectClassHandle whichClass) // supplied C1
throw (
  ObjectClassNotDefined,
  AttributeNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.17
ObjectClassHandle            // returned C3
getObjectClass (
  ObjectHandle theObject) // supplied C1
throw (
  ObjectNotKnown,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.18
SpaceHandle                  // returned C3
getInteractionRoutingSpaceHandle (
  InteractionClassHandle theHandle) // supplied C1
throw (
  InteractionClassNotDefined,
  FederateNotExecutionMember,
  ConcurrentAccessAttempted,
  RTIinternalError);

// 10.19
TransportationHandle         // returned C3
getTransportationHandle (
  const char *theName) // supplied C4

```

```

throw (
    NameNotFound,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.20
char *                               // returned C6
getTransportationName (
    TransportationHandle theHandle) // supplied C1
throw (
    InvalidTransportationHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.21
OrderingHandle           // returned C3
getOrderingHandle (
    const char *theName) // supplied C4
throw (
    NameNotFound,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.22
char *                               // returned C6
getOrderingName (
    OrderingHandle theHandle) // supplied C1
throw (
    InvalidOrderingHandle,
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

// 10.23
void enableClassRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.24
void disableClassRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.25
void enableAttributeRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.26
void disableAttributeRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.27
void enableAttributeScopeAdvisorySwitch()

```

```

throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.28
void disableAttributeScopeAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.29
void enableInteractionRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 10.30
void disableInteractionRelevanceAdvisorySwitch()
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    SaveInProgress,
    RestoreInProgress,
    RTIinternalError);

// 
Boolean // returned C3
tick ()
throw (
    SpecifiedSaveLabelDoesNotExist,
    ConcurrentAccessAttempted,
    RTIinternalError);

Boolean           // returned C3
tick (
    TickTime minimum, // supplied C1
    TickTime maximum) // supplied C1
throw (
    SpecifiedSaveLabelDoesNotExist,
    ConcurrentAccessAttempted,
    RTIinternalError);

RTIambassador()
throw (
    MemoryExhausted,
    RTIinternalError);

~RTIambassador()
throw (RTIinternalError);

RegionToken
getRegionToken(
    Region *)
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RegionNotKnown,
    RTIinternalError);

Region *
getRegion(
    RegionToken)
throw(
    FederateNotExecutionMember,
    ConcurrentAccessAttempted,
    RTIinternalError);

```

```
RegionNotKnown,  
RTIinternalError);
```

```

//File federateAmbServices.hh
//Included in RTI.hh

//          RTI Parameter Passing Memory Conventions
//
// C1  In parameter by value.
// C2  Out parameter by pointer value.
// C3  Function return by value.
// C4  In parameter by const pointer value.  Caller provides memory.
//      Caller may free memory or overwrite it upon completion of
//      the call.  Callee must copy during the call anything it
//      wishes to save beyond completion of the call.  Parameter
//      type must define const accessor methods.
// C5  Out parameter by pointer value.  Caller provides reference to object.
//      Callee constructs an instance on the heap (new) and returns.
//      The caller destroys the instance (delete) at its leisure.
// C6  Function return by pointer value.  Callee constructs an instance on
//      the heap (new) and returns a reference.  The caller destroys the
//      instance (delete) at its leisure.
//

///////////////////////////////
// Federation Management Services //
///////////////////////////////

// 4.7
virtual void synchronizationPointRegistrationSucceeded (
    const char *label) // supplied C4)
throw (
    FederateInternalError) = 0;

virtual void synchronizationPointRegistrationFailed (
    const char *label) // supplied C4)
throw (
    FederateInternalError) = 0;

// 4.8
virtual void announceSynchronizationPoint (
    const char *label, // supplied C4
    const char *tag) // supplied C4
throw (
    FederateInternalError) = 0;

// 4.10
virtual void federationSynchronized (
    const char *label) // supplied C4)
throw (
    FederateInternalError) = 0;

// 4.12
virtual void initiateFederateSave (
    const char *label) // supplied C4
throw (
    UnableToPerformSave,
    FederateInternalError) = 0;

// 4.15
virtual void federationSaved ()
throw (
    FederateInternalError) = 0;

virtual void federationNotSaved ()
throw (
    FederateInternalError) = 0;

// 4.17
virtual void requestFederationRestoreSucceeded (
    const char *label) // supplied C4
throw (
    FederateInternalError) = 0;

virtual void requestFederationRestoreFailed (
    const char *label, // supplied C4
    const char *reason) // supplied C4

```

```

throw (
    FederateInternalError) = 0;

// 4.18
virtual void federationRestoreBegun ()
throw (
    FederateInternalError) = 0;

// 4.19
virtual void initiateFederateRestore (
    const char *label, // supplied C4
        FederateHandle handle) // supplied C1
throw (
    SpecifiedSaveLabelDoesNotExist,
    CouldNotRestore,
    FederateInternalError) = 0;

// 4.21
virtual void federationRestored ()
throw (
    FederateInternalError) = 0;

virtual void federationNotRestored ()
throw (
    FederateInternalError) = 0;

///////////////////////////////
// Declaration Management Services //
///////////////////////////////

// 5.10
virtual void startRegistrationForObjectClass (
    ObjectClassHandle theClass) // supplied C1
throw (
    ObjectClassNotPublished,
    FederateInternalError) = 0;

// 5.11
virtual void stopRegistrationForObjectClass (
    ObjectClassHandle theClass) // supplied C1
throw (
    ObjectClassNotPublished,
    FederateInternalError) = 0;

// 5.12
virtual void turnInteractionsOn (
    InteractionClassHandle theHandle) // supplied C1
throw (
    InteractionClassNotPublished,
    FederateInternalError) = 0;

// 5.13
virtual void turnInteractionsOff (
    InteractionClassHandle theHandle) // supplied C1
throw (
    InteractionClassNotPublished,
    FederateInternalError) = 0;

/////////////////////////////
// Object Management Services //
/////////////////////////////

// 6.3
virtual void discoverObjectInstance (
    ObjectHandle theObject, // supplied C1
    ObjectClassHandle theObjectClass) // supplied C1
throw (
    CouldNotDiscover,
    ObjectClassNotKnown,
    FederateInternalError) = 0;

// 6.5
virtual void reflectAttributeValue (
    ObjectHandle theObject, // supplied C1

```

```

const AttributeHandleValuePairSet& theAttributes, // supplied C4
const FedTime& theTime, // supplied C1
const char *theTag, // supplied C4
EventRetractionHandle theHandle) // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateOwnsAttributes,
    InvalidFederationTime,
    FederateInternalError) = 0;

virtual void reflectAttributeValues (
    ObjectHandle theObject, // supplied C1
    const AttributeHandleValuePairSet& theAttributes, // supplied C4
    const char *theTag) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateOwnsAttributes,
    FederateInternalError) = 0;

// 6.7
virtual void receiveInteraction (
    InteractionClassHandle theInteraction, // supplied C1
    const ParameterHandleValuePairSet& theParameters, // supplied C4
    const FedTime& theTime, // supplied C4
    const char *theTag, // supplied C4
    EventRetractionHandle theHandle) // supplied C1
throw (
    InteractionClassNotKnown,
    InteractionParameterNotKnown,
    InvalidFederationTime,
    FederateInternalError) = 0;

virtual void receiveInteraction (
    InteractionClassHandle theInteraction, // supplied C1
    const ParameterHandleValuePairSet& theParameters, // supplied C4
    const char *theTag) // supplied C4
throw (
    InteractionClassNotKnown,
    InteractionParameterNotKnown,
    FederateInternalError) = 0;

// 6.9
virtual void removeObjectInstance (
    ObjectHandle theObject, // supplied C1
    const FedTime& theTime, // supplied C4
    const char *theTag, // supplied C4
    EventRetractionHandle theHandle) // supplied C1
throw (
    ObjectNotKnown,
    InvalidFederationTime,
    FederateInternalError) = 0;

virtual void removeObjectInstance (
    ObjectHandle theObject, // supplied C1
    const char *theTag) // supplied C4
throw (
    ObjectNotKnown,
    FederateInternalError) = 0;

// 6.13
virtual void attributesInScope (
    ObjectHandle theObject, // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateInternalError) = 0;

// 6.14
virtual void attributesOutOfScope (
    ObjectHandle theObject, // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4

```

```

throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateInternalError) = 0;

// 6.16
virtual void provideAttributeValueUpdate (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeNotOwned,
    FederateInternalError) = 0;

// 6.17
virtual void turnUpdatesOnForObjectInstance (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotOwned,
    FederateInternalError) = 0;

// 6.18
virtual void turnUpdatesOffForObjectInstance (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotOwned,
    FederateInternalError) = 0;

///////////////////////////////
// Ownership Management Services //
///////////////////////////////

// 7.4
virtual void requestAttributeOwnershipAssumption (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& offeredAttributes, // supplied C4
    const char          *theTag)           // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeAlreadyOwned,
    AttributeNotPublished,
    FederateInternalError) = 0;

// 7.5
virtual void attributeOwnershipDivestitureNotification (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& releasedAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeNotOwned,
    AttributeDivestitureWasNotRequested,
    FederateInternalError) = 0;

// 7.6
virtual void attributeOwnershipAcquisitionNotification (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& securedAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeAcquisitionWasNotRequested,
    AttributeAlreadyOwned,
    AttributeNotPublished,
    FederateInternalError) = 0;

// 7.9
virtual void attributeOwnershipUnavailable (

```

```

        ObjectHandle      theObject,          // supplied C1
        const AttributeHandleSet& theAttributes) // supplied C4
    throw (
        ObjectNotKnown,
        AttributeNotKnown,
        AttributeAlreadyOwned,
        AttributeAcquisitionWasNotRequested,
        FederateInternalError) = 0;

// 7.10
virtual void requestAttributeOwnershipRelease (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& candidateAttributes, // supplied C4
    const char         *theTag)             // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeNotOwned,
    FederateInternalError) = 0;

// 7.14
virtual void confirmAttributeOwnershipAcquisitionCancellation (
    ObjectHandle      theObject,          // supplied C1
    const AttributeHandleSet& theAttributes) // supplied C4
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    AttributeAlreadyOwned,
    AttributeAcquisitionWasNotCanceled,
    FederateInternalError) = 0;

// 7.16
virtual void informAttributeOwnership (
    ObjectHandle      theObject,          // supplied C1
    AttributeHandle   theAttribute,        // supplied C1
    FederateHandle   theOwner)           // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateInternalError) = 0;

virtual void attributeIsNotOwned (
    ObjectHandle      theObject,          // supplied C1
    AttributeHandle   theAttribute)       // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateInternalError) = 0;

virtual void attributeOwnedByRTI (
    ObjectHandle      theObject,          // supplied C1
    AttributeHandle   theAttribute)       // supplied C1
throw (
    ObjectNotKnown,
    AttributeNotKnown,
    FederateInternalError) = 0;

///////////////////////////////
// Time Management Services //
///////////////////////////////

// 8.3
virtual void timeRegulationEnabled (
    const FedTime& theFederateTime) // supplied C4
throw (
    InvalidFederationTime,
    EnableTimeRegulationWasNotPending,
    FederateInternalError) = 0;

// 8.6
virtual void timeConstrainedEnabled (
    const FedTime& theFederateTime) // supplied C4
throw (
    InvalidFederationTime,

```

```
EnableTimeConstrainedWasNotPending,
FederateInternalError) = 0;

// 8.13
virtual void timeAdvanceGrant (
    const FedTime& theTime) // supplied C4
throw (
    InvalidFederationTime,
    TimeAdvanceWasNotInProgress,
    FederateInternalError) = 0;

// 8.22
virtual void requestRetraction (
    EventRetractionHandle theHandle) // supplied C1
throw (
    EventNotKnown,
    FederateInternalError) = 0;
```